

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application for Letters Patent

Title : TRANSMITTING APPARATUS, TRANSMITTING
METHOD, RECEIVING APPARATUS, RECEIVING
METHOD, TRANSMITTING AND RECEIVING
SYSTEM, AND TRANSMITTING AND RECEIVING
METHOD

Inventor(s): Yasuaki YAMAGISHI
Kazuhiko TAKABAYASHI
Kazuo HARAOKA
Yoshihisa GONNO
Fumihiko NISHIO

TRANSMITTING APPARATUS, TRANSMITTING METHOD, RECEIVING APPARATUS, RECEIVING METHOD, TRANSMITTING AND RECEIVING SYSTEM, AND TRANSMITTING AND RECEIVING METHOD

BACKGROUND OF THE INVENTION

5 Field of the Invention

The present invention relates to a transmitting apparatus, a transmitting method, a receiving apparatus, a receiving method, a transmitting and receiving apparatus, and a transmitting and receiving method suitable for broadcasting hierarchical data that is distributed on a network.

10 Description of the Related Art

Many data delivering methods have been proposed. For example, http (Hyper Text Transfer Protocol) is used to publish pages on the Internet. TCP/IP (Transmission Control Protocol/Internet Protocol) enables computers connected to the Internet to exchange data. In the TCP/IP, a receiving side that receives data calls a transmitting side. Whenever data is transmitted or received corresponding to the TCP/IP, a connection is established between the receiving side and the transmitting side. Thus, with such protocols, data can be delivered with high reliability. On the other hand, the transmitting side and the network are adversely loaded and thereby data cannot be effectively delivered.

In other words, when the number of terminal

units that receive data is becoming large and they access a server that delivers data thereto at a time, the server and the network are adversely loaded. Thus, even if a terminal unit requests the server for data, the terminal unit cannot receive the requested data from the server in a short time.

To solve such a problem, a method using a satellite line, a CATV (Cable Television) line, a ground wave digital broadcast, and so forth that allows data to be broadcast has been proposed. With this method, even if the number of terminal units increases, the server and the network are prevented from being adversely overloaded.

In recent years, as digital communication networks such as the Internet have become common, a huge amount of data has been stored on the networks. Thus, it is desired to effectively use such data. To do that, a directory service for hierarchically managing data distributed on a network and providing the data to the user is becoming popular. Using the directory service, the user can quickly find desired information from data distributed on the network and access the desired information.

The directory service has been set forth as X.500 series in OSI (Open System Interconnection) that is an international standard. In the X.500, the directory is defined as a set of open systems.

Individual open systems cooperatively have logical databases of information with respect to a set of objects of the real world.

With major directory services defined in the X.500, the user can search and browse information stored in the directory. The directory services also provide the user with a list service (such as a telephone directory) and a user authenticating service. In the directory service, each object is assigned a unique name so that the user can easily memorize, infer, and recognize each object.

The directory services defined in the X.500 are very comprehensive. The program size of each directory service is very large. Thus, it is very difficult to accomplish a directory service on the Internet that uses the TCP/IP as a protocol. To solve such a problem, LDAP (Lightweight Directory Access Protocol) has been proposed as a compact type directory service for the TCP/IP.

In recent years, a directory service using a broadcast data transmitting means such as a satellite line, a CATV line, a ground digital broadcast, or the like has been proposed. In this case, information is uni-directionally delivered with the directory service. Thus, the user cannot request the directory service for desired data. Consequently, in such a directory service, the same information is repeatedly

transmitted. The user side stores received information to an IRD (Integrated Receiver Decoder) or an STB (Set Top Box) that is a digital broadcast receiver connected to a television receiver.

5 Next, information synchronization between the directory server side and the user side (namely, synchronization management) will be considered. The directory server side detects a substantial update of the directory corresponding to a dynamic change of the hierarchical structure of the directory. The directory server side transmits directory information corresponding to the change of the hierarchical structure of the directory to the user. At this point, only the difference of the changed data rather than all directory information is extracted and transmitted.

10 The directory server side transmits difference update data to the user whenever the directory server side detects an update of the directory information or at intervals of a predetermined time period. When the user side always receives the difference update data corresponding to an update of the directory and updates the stored directory information with the received difference update data, the directory information is synchronized between the directory server and the user.

25 Now, it is assumed that the user side is in a pause state (for example, the power of the receiver is

turned off) and cannot receive difference update data from the directory server for a predetermined time period. For example, as shown in Fig. 1, the directory server successively transmits difference update data
5 Msg1, Msg2, and Msg3 to the user. The difference update data Msg2 is the difference of the directory information updated by the difference update data Msg1. The difference update data Msg3 is the difference of the directory information updated by the difference
10 update data Msg2.

To solve such a problem, for example, an overall structure of a directory is transmitted from the directory server to the user side at intervals of a predetermined time period. Difference update data
15 against the received overall structure is transmitted in the above described manner. However, when the overall directory structure is frequently transmitted, the transmitting resources are wastefully used. Thus, in the conventional method, the advantageous feature of
20 which difference update data is transmitted becomes meaningless.

OBJECTS AND SUMMARY OF THE INVENTION

Therefore, an object of the present invention is to provide a transmitting apparatus, a transmitting
25 method, a receiving apparatus, a receiving method, a transmitting and receiving apparatus, and a transmitting and receiving method that allow an overall

directory structure to be broadcast without a loss of transmitting resources so as to update directory information without a mismatch between the transmitting side and the receiving side even if the receiving side cannot temporarily receive difference update data.

5 A first aspect of the present invention is a transmitting apparatus, comprising a transmitting means for transmitting the same contents of update information a plurality number of times, the update information representing that data has been updated, wherein the transmission timing of the same contents of the update information transmitted by the transmitting means is designated corresponding to operating state information of receiving means that receives the update information from the transmitting means.

10 A second aspect of the present invention is a transmitting method, comprising the step of a transmitting the same contents of update information a plurality number of times, the update information representing that data has been updated, wherein the transmission timing of the same contents of the update information transmitted at the transmitting step is designated corresponding to operating state information of receiving means that receives the update information from the transmitting means.

20 A third aspect of the present invention is a receiving apparatus for receiving a hierarchical

structure of a directory for hierarchically managing the locations of contents data that is transmitted, comprising a receiving means for receiving first difference information, second difference information, and third difference information, the first difference information being obtained by detecting a change of container entries, the second difference information being obtained by detecting a change of leaf entries, the directory being composed of container entries and leaf entries, a container entry containing information in the immediately lower hierarchical level thereof, a leaf entry not containing information in the immediately lower hierarchical level thereof, the third difference information being transmitted at a predetermined time period, the third difference information being capable of reproducing the hierarchical structure of the directory, and a managing means for managing the hierarchical structure of the directory formed corresponding to the first difference information, the second difference information, and the third difference information received by the receiving means, wherein local operating state information of the receiving apparatus is monitored and transmitted.

A fourth aspect of the present invention is a receiving method for receiving a hierarchical structure of a directory for hierarchically managing the locations of contents data that is transmitted,

comprising the steps of receiving first difference information, second difference information, and third difference information, the first difference information being obtained by detecting a change of container entries, the second difference information being obtained by detecting a change of leaf entries, the directory being composed of container entries and leaf entries, a container entry containing information in the immediately lower hierarchical level thereof, a leaf entry not containing information in the immediately lower hierarchical level thereof, the third difference information being transmitted at a predetermined time period, the third difference information being capable of reproducing the hierarchical structure of the directory, and managing the hierarchical structure of the directory formed corresponding to the first difference information, the second difference information, and the third difference information received at the receiving step, wherein local operating state information of the receiving apparatus is monitored and transmitted.

A fifth aspect of the present invention is a transmitting and receiving system, comprising a transmitting means for transmitting the same contents of update information a plurality number of times, the update information representing that data has been updated, and a receiving means for receiving the update

information transmitted by the transmitting means,
wherein the transmission timing of the same contents of
the update information transmitted by the transmitting
means is designated corresponding to operating state
5 information of the receiving means.

A sixth aspect of the present invention is a
transmitting and receiving method, comprising the step
of transmitting the same contents of update information
a plurality number of times, the update information
10 representing that data has been updated, and receiving
the update information transmitted at the transmitting
step, wherein the transmission timing of the same
contents of the update information transmitted at the
transmitting step is designated corresponding to
15 operating state information received at the receiving
step.

As was described above, according to the
first aspect and the second aspect of the present
invention, the same contents of update information is
20 transmitted a plurality number of times, the update
information representing that data has been updated.
The transmission timing of the same contents of the
transmitted update information is designated
corresponding to operating state information of a
25 receiving means. Thus, the reception probability of
which the receiving side receives update information is
high.

According to the third aspect and the fourth aspect of the present invention, first difference information, second difference information, and third difference information are received. The first difference information is obtained by detecting a change of container entries. The second difference information is obtained by detecting a change of leaf entries. The directory is composed of container entries and leaf entries. A container entry contains information in the immediately lower hierarchical level thereof. A leaf entry does not contain information in the immediately lower hierarchical level thereof. The third difference information is transmitted at a predetermined time period. The third difference information can be used to reproduce the hierarchical structure of the directory. The hierarchical structure of the directory formed is managed corresponding to the first difference information, the second difference information, and the third difference information that have been received. Local operating state information of the receiving apparatus is monitored and transmitted. Thus, the operating state information can be affected to the third difference information transmitted by the transmitting side at the intervals of the predetermined time period,

According to the fifth aspect and the sixth aspect of the present invention, the transmitting side

transmits the same contents of update information a plurality number of times, the update information representing that data has been updated. The receiving side receives the update information that has been transmitted. The transmission timing of the same contents of the update information that is transmitted is designated corresponding to operating state information on the receiving side. Thus, the update information is transmitted corresponding to the operating state of the receiving side.

These and other objects, features and advantages of the present invention will become more apparent in light of the following detailed description of a best mode embodiment thereof, as illustrated in the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a schematic diagram showing an example of which a receiver side cannot receive difference update data from a directory server for a predetermined time period;

Fig. 2 is a schematic diagram showing a system according to the present invention;

Fig. 3 is a schematic diagram for explaining a system of which a plurality of receiving sides are connected to a broadcasting network;

Fig. 4 is a schematic diagram for explaining a directory structure;

Fig. 5 is a schematic diagram showing an example of the structure of a container entry;

Figs. 6A and 6B are schematic diagrams showing an example of the structure of a leaf entry;

5 Fig. 7 is a functional block diagram for explaining the function of a transmitting side replicater;

Fig. 8 is a functional block diagram for explaining the function of a receiving side client;

10 Fig. 9 is a functional block diagram for explaining the function of a receiving side server;

Figs. 10A, 10B, 10C, 10D, 10E, and 10F are schematic diagrams for explaining difference update information of a directory structure;

15 Figs. 11A, 11B, 11C, and 11D are schematic diagrams for explaining difference update information of a directory structure;

Fig. 12 is a schematic diagram for explaining overall structure information of a directory structure;

20 Fig. 13 is a schematic diagram for explaining overall structure information of a directory structure;

Fig. 14 is a flow chart for explaining a synchronization managing method of container entries;

25 Figs. 15A and 15B are flow charts for explaining the synchronization managing method of the container entries in detail;

Fig. 16 is a flow chart for explaining the

synchronization managing method of the container entries in detail;

Fig. 17 is a flow chart for explaining a synchronization managing method of leaf entries;

5 Figs. 18A and 18B are flow charts for explaining the synchronization managing method of the leaf entries in detail;

10 Fig. 19 is a flow chart for explaining the synchronization managing method of the leaf entries in detail;

Figs. 20A, 20B, 20C, and 20D are schematic diagrams for explaining a bit arrangement structure of mask values of filtering masks;

15 Fig. 21 is a flow chart showing a mask value assigning process corresponding to the increase/decrease of a container entry in the case that an entry is added or deleted;

Figs. 22A and 22B are schematic diagrams for explaining a container entry mask scheme that is coded;

20 Figs. 23A and 23B are schematic diagrams for explaining a target mask list;

Fig. 24 is a flow chart showing a process for generating a target mask list;

25 Fig. 25 is a flow chart showing a process for selectively receiving broadcast leaf update information Msg.x1' corresponding to a target mask list;

Fig. 26 is a flow chart showing an example of

a process for setting a value of an overall structure information notification interval timer; and

Fig. 27 is a schematic diagram showing the result of an example of a statistic process for an operating ratio on the receiving side.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Next, with reference to the accompanying drawings, an embodiment of the present invention will be described. Fig. 2 shows an example of the structure of a system according to the present invention. A transmitting side 1 arranges many contents data distributed on a network such as the Internet or a broadcast network in a tree shape hierarchical structure and manages it as a directory structure. The transmitting side 1 transmits directory information that represents the directory structure to a broadcasting network 2. As shown in Fig. 3, many receiving devices on the receiving side 3 are connected to the broadcast network 2. The receiving side 3 can receive broadcast programs through the broadcasting network 2. The receiving side 3 receives directory information that is broadcast on the broadcasting network 2. With reference to the received directory information, a receiving side replicater 17 (that will be described later) on the receiving side 3 can select required information title from many information titles distributed on the broadcasting network 2 and other

networks and obtain the selected information title.

The transmitting side 1 and receiving side 3 are connected to each other through a bidirectional network 4. As shown in Fig. 3, many receiving devices on the receiving side 3 are connected to the bidirectional network 4 so that the receiving side devices bidirectionally communicate with the transmitting side 1.

As shown in Fig. 2, the transmitting side 1 comprises a transmitting side directory service client 10 (hereinafter referred to as transmitting side client 10), a transmitting side directory server 11 (hereinafter referred to as transmitting side server 11), and a transmitting side directory server replicater 12 (hereinafter referred to as transmitting side replicater 12). The transmitting side client 10, the transmitting side server 11, and the transmitting side replicater 12 are connected with a network such as the Internet so that they communicate with each other.

The transmitting side client 10 is a contents data provider that provides the user with contents data through a network (not shown). The transmitting side client 10 changes and updates the directory structure. The transmitting side client 10 may be disposed at any location of the network. The transmitting side server 11 inquires and changes the contents of the transmitting side client 10 and manages the directory

structure. Many transmitting side servers 11 may be distributed on the network. The transmitting side replicater 12 monitors the directory structure managed by the transmitting side server 11 and detects an
5 update of the directory structure. The transmitting side replicater 12 compares the pre-updated directory structure with the post-updated directory structure corresponding to the detected result, extracts the difference thereof, and generates the difference update
10 information of the directory structure. The structure of the difference update information will be described later.

The difference update information is periodically transmitted to the broadcasting network 2.
15 In addition, according to the present invention, the overall information of the directory structure managed by the transmitting side server 11 is transmitted to the broadcasting network 2 (hereinafter, this overall information is referred to as overall structure
20 information). As will be described later, the transmission intervals of the difference update information and those of the overall structure information are separately set. The receiving side 3 forms a local directory structure corresponding to the
25 received difference update information and overall structure information.

The receiving side 3 comprises a receiving

side directory server replicater 17 (hereinafter referred to as receiving side replicater 17), a receiving side directory server 16 (hereinafter referred to as receiving side server 16), and a receiving side directory service client 15 (hereinafter referred to as receiving side client 15). The receiving side 3 is for example a personal computer, an STB, or an IRD (as were described in the section of "Description of the Related Art"). The receiving side client 15 is for example application software such as WWW (World Wide Web) browser that accesses a directory structure and obtains and displays a plurality of different formats of data. The receiving side server 16 is for example a local database that stores directory information.

The overall structure information, the update information of the directory structure, and the difference information of the update information that are transmitted through the broadcasting network 2 are received by the receiving side replicater 17. The receiving side replicater 17 updates the local database stored in the receiving side server 16 corresponding to the received information and reforms the directory structure. Corresponding to a user's request or the like, the receiving side client 15 requests the receiving side replicater 17 for desired information. Corresponding to the request, the receiving side

replicater 17 searches the database of the receiving side server 16 and returns an address of the required information to the receiving side client 15.

Corresponding to the returned address, the receiving side client 15 can access information distributed on the network (not shown).

On the other hand, the transmitting side replicater 12 and the receiving side replicater 17 are connected to each other with the bidirectional network 4. Many receiving side replicaters 17 are connected to the bidirectional network 4. At least one receiving side replicater 17 bidirectionally communicates with the transmitting side replicater 12 through the bidirectional network 4.

Next, with reference to Fig. 4, the directory structure will be described. As shown in Fig. 4, the directory is hierarchically structured in a tree shape. Each node of the tree is referred to as entry. Each entry contains information. There are three types of entries that are one root entry, a plurality of container entries, and a plurality of leaf entries. A container entry can contain an entry in the immediately lower hierarchical level thereof. A hierarchy formed with container entries is hereinafter referred to as container hierarchy.

Entries other than the root entry and container entries are referred to as leaf entries. A

leaf entry cannot contain an entry in the immediately lower hierarchical level thereof. Thus, a leaf entry is a terminal node that cannot contain an entry. A hierarchy of a leaf entry is hereinafter referred to leaf hierarchy. A leaf hierarchy is contained in a container entry.

An entry in the highest hierarchical level of the directory tree is referred to as root entry. The root entry represents the entire world of the directory structure. In the following description, it is assumed that a container entry contains at least one leaf entry or at least one container entry.

Each entry has a plurality of attributes. A name that is uniquely identified in the directory tree is referred to as entry name. With an entry name, the location of the entry in the directory structure can be designated. In the example shown in Fig. 4, the root entry is assigned an entry name A. A leaf entry in the immediately lower hierarchical level of the root entry (an entry at the lower left position of the root entry) is assigned an entry name A.B. A container entry in the immediately lower hierarchical level (an entry at the lower right position of the root direction) is assigned an entry name A.C. Each entry is assigned an entry name with periods corresponding to lower hierarchical levels routed from the root entry.

Fig. 5 shows an example of the structure of a

container entry. A container entry has attributes thereof, a list of container entries in the immediately lower hierarchical level of the current container entry, and a list of leaf entries in the immediately lower hierarchical level of the current container entry. The list of entries in the immediately lower hierarchical level of the current container entry may not contain elements. The number of attributes of the current container entry may be two or more as shown in Fig. 5.

Figs. 6A and 6B show an example of the structure of a leaf entry. As shown in Fig. 6A, a leaf entry has a plurality of attributes. Fig. 6B shows a real example of attributes of a leaf entry. Each attribute is composed of an attribute name and an attribute value. When a leaf entry is search information of contents data, one attribute name is an address. The attribute value is address information of contents data such as URL (Uniform Resource Locator).

The directory structure has container entries arranged in a tree shape corresponding to information genres under the root entry that represents the entire world.

Next, the structure of the transmitting side 1 will be described in detail. The transmitting side server 11 manages the directory structure as with the structure shown in Figs. 4, 5, 6A, and 6B. The

transmitting side client 10 changes the directory structure managed by the transmitting side server 11 corresponding to contents data that the transmitting side client 10 provides. A change of contents data of the transmitting side server 11 is monitored by the transmitting side replicater 12.

Fig. 7 is a functional block diagram for explaining the function of the transmitting side replicater 12. The transmitting side replicater 12 can be composed of for example a conventional computer system. The transmitting side replicater 12 comprises a CPU (Central Processing Unit), recording and storing mediums (such as a memory and a hard disk), a communicating means, and a user interface. The functional block shown in Fig. 7 is accomplished by application software that runs on the CPU. Each module shown in Fig. 7 is a functional element of the application software.

The transmitting side replicater 12 comprises an update detecting module 20, a message generating module 21, and a message broadcasting module 22. Each of the update detecting module 20, the message generating module 21, and the message broadcasting module 22 has a module that performs a process for a container hierarchy and a module that performs a process for a leaf hierarchy.

The update detecting module 20 is a module

that references the transmitting side server 11 and detects whether or not the directory structure managed by the transmitting side server 11 has been changed. The update detecting module 20 is composed of a
5 container hierarchy update detecting module 23 and a leaf hierarchy update detecting module 24. The container hierarchy update detecting module 23 monitors the transmitting side server 11 and detects whether or not the structure of the container hierarchy has been
10 changed. The leaf hierarchy update detecting module 24 monitors the transmitting side server 11 and detects whether the structure of the leaf hierarchy and the contents of the leaf entry have been changed.

The message generating module 21 is a module
15 that generates a message that represents difference update information of the directory structure corresponding to the detected result of the change of the directory structure by the update detecting module 20. The message generating module 21 is composed of a
20 container structure update message generating module 25 and a leaf entry update message generating module 26. The container structure update message generating module 25 generates a message that represents
25 difference update information of the structure change of the container hierarchy corresponding to the detected result of the container hierarchy update detecting module 23. The leaf entry update message

generating module 26 generates a message that represents update information of the leaf hierarchy corresponding to the detected result of the leaf hierarchy update detecting module 24.

5 The message broadcasting module 22 is a module that broadcasts the message generated by the message generating module 21 to the broadcasting network 2. The message broadcasting module 22 is composed of a container structure update message
10 broadcasting module 27 and a leaf entry update message broadcasting module 28. The container structure update message broadcasting module 27 broadcasts the message generated by the container structure update message
15 generating module 25. The leaf entry update message broadcasting module 28 broadcasts the message generated by the leaf entry update message generating module 26. The message broadcasting module 22 cyclically
20 broadcasts the same message to the broadcasting network 2 a predetermined number of times.

25 Next, the structure of the receiving side 3 will be described in more reality. Fig. 8 is a functional block diagram for explaining the function of the receiving side client 15. The receiving side client 15 can be composed of a conventional computer
30 system. The receiving side client 15 comprises a CPU (Central Processing Unit), recording and storing mediums (such as a memory and a hard disk), a

communicating means, and a user interface. The functional block shown in Fig. 8 is accomplished by application software that runs on the CPU. Each module is a functional element of the application software.

5 As was described above, the receiving side client 15 is for example a WWW browser. The receiving side client 15 can integrally display and reproduce supplied contents data (for example still picture data, text data, audio data, and moving picture data). In addition, corresponding to a user's request that is input with a predetermined inputting means, the receiving side client 15 can control the displaying operation and the reproducing operation of such data.

10 The receiving side client 15 comprises a directory searching module 30, a user interactive managing module 31, and a contents data obtaining module 32. A user interface 33 is connected to the user interactive managing module 31. The user interface 33 is composed of a text inputting means (such as a keyboard), a pointing device (such as a mouse), and a displaying device. A contents data search request to the receiving side client 15 is interactively performed with the user interactive managing module 31 through the user interface 33.

20 When a contents data search request is input to the user interactive managing module 31, the user interactive managing module 31 requests the directory

searching module 30 for a directory entry corresponding to the desired contents data so as to search the address of the desired contents data. Corresponding to the search request, the directory searching module 30
5 . transmits a directory entry search request to the receiving side server 16.

The search result of the directory entry corresponding to the search request is returned from the receiving side server 16 to the directory searching
10 module 30. The search result is further returned from the directory searching module 30 to the user interactive managing module 31. When the directory entry information of the search result represents that the search result is a leaf entry, the address
15 information of the contents data is extracted as one attribute. The user interactive managing module 31 transmits a contents data obtaining request to the contents data obtaining module 32 so as to obtain contents data corresponding to the extracted address
20 information.

Corresponding to the received contents data obtaining request, the contents data obtaining module 32 transmits a contents data obtaining request to a contents data server 35. The contents data server 35
25 is a server connected to the receiving side client 15 through a bidirectional network 36 such as the Internet. The contents data server 35 provides the

user with contents data. The contents data may be provided through the bidirectional network 36 or the broadcasting network 2.

The contents data obtained from the contents data server 35 corresponding to the contents data obtaining request is supplied to the contents data obtaining module 32 through for example the bidirectional network 36. The contents data is returned from the contents data obtaining module 32 to the user interactive managing module 31. The user interactive managing module 31 outputs the received contents data to the user interface 33.

When the requested contents data is transmitted through the broadcasting network 2, the contents data obtaining module 32 may directly obtain desired contents data that is broadcast through the broadcasting network 2 corresponding to the contents data obtaining request.

Fig. 9 is a functional block diagram for explaining the function of the receiving side server 16. As with the receiving side client 15, the receiving side server 16 is composed of a conventional computer system. The receiving side server 16 comprises a directory update request processing module 40, a directory database 41, and a directory search request processing module 42.

The directory database 41 stores directory

information corresponding to the directory structure managed by the transmitting side server 11. As was described above, the receiving side replicater 17 receives difference update information of the directory structure from the transmitting side 1 through the broadcasting network 2. Although details will be described later, the receiving side replicater 17 transmits a request to the directory update request processing module 40 so as to update the directory information stored in the directory database 41 corresponding to the difference update information. Corresponding to the request, the directory update request processing module 40 updates the directory information stored in the directory database 41 with the difference update information.

On the other hand, the search request for the directory entry transmitted from the receiving side client 15 is received by the directory search request processing module 42. The directory search request processing module 42 searches the directory database 41 for the required directory entry corresponding to the received search request. The directory entry as the search result (for example, address information of a leaf entry) is returned from the directory search request processing module 42 to the receiving side client 15.

Since the system is structured as described

above, the user can search directory information with the receiving side client 15 and obtain address information with desired contents data as the search result. The user can obtain desired contents data corresponding to the obtained address information. The directory structure is monitored by the transmitting side replicater 12. The difference update information and the overall structure information of the directory structure are transmitted at intervals of a predetermined time period and supplied to the receiving side replicater 17 through the broadcasting network 2. Corresponding to the supplied difference update information and overall structure information, the receiving side replicater 17 on the user side updates the directory information stored in the directory database 41. Thus, the user can always have directory information that synchronizes with a real directory structure in the directory database 41.

Next, with reference to Figs. 10A, 10B, 10C, 10D, 10E, 10F, 11A, 11B, 11C, 11D, 12, and 13, the difference update information and the overall structure information of the directory structure will be described. In the following description, a process for adding or deleting a container entry C or a leaf entry 1 to/from the immediately lower hierarchical level of a container entry X of a particular container hierarchy designated by a schema version Sv is expressed as

follows.

(Sv, X, [+/-] [C/1])

The expression of the process for the directory structure represents the difference between the pre-processed directory structure and the post-processed directory structure. The expression can be used as difference update information.

On the other hand, the overall structure information is information that represents a directory structure at a particular time point. Assuming that the overall structure information is the difference to the root entry, the overall structure information can be represented with the above-described difference update information.

The schema version Sv is a value that changes corresponding to the change of the directory structure. The container entry X (or C) is a container entry name. In this example, a container entry name is represented by an uppercase alphabet character. The leaf entry 1 represents a leaf entry name. In this example, a leaf entry name is represented by a lowercase alphabet character. An addition of an entry is represented by [+]. A deletion of an entry is represented by [-]. A slash mark in parentheses [] represents that one of two characters therein is placed. In Figs. 10A to 10F and 11A to 11D, a concentric square represents a container entry, whereas a single square represents a leaf entry.

In Figs. 10A to 10F and 11A to 11D, the root entry is not shown except for a connection line thereof.

Fig. 10A shows a directory structure 100 of which only a container entry A is disposed in the immediately lower hierarchical level of the root entry (not shown). This state is referred to as schema version $Sv = 1$. In this state, a process of $(1, A, +B)$ is performed. In other words, a container entry B is added to the immediately lower hierarchical level of the container entry A. Thus, a directory structure 101 as shown in Fig. 10B is generated. Since the container entry B is added to the state shown in Fig. 10A, since a hierarchical image of the container entries is changed, the schema version is changed to $Sv = 2$.

In the state shown in Fig. 10B, a process of $(2, A, +a)$ is performed. In other words, a leaf entry a is added to the immediately lower hierarchical level of the container entry A. Thus, a directory structure 102 shown in Fig. 10C is generated. In addition, a process of $(2, A, -a)$ is performed in the state shown in Fig. 10C. In other words, the leaf entry a is deleted from the immediately lower hierarchical level of the container entry A. Thus, a directory structure 103 as shown in Fig. 10D is generated. Thereafter, a process of $(2, A, -B)$ is performed in the state shown in Fig. 10D. In other words, the container entry B is deleted from the immediately lower hierarchical level

of the container entry A. Thus, a directory structure 104 shown in Fig. 10E is generated.

5 In the state shown in Fig. 10E, since the hierarchical image of the container entry has been changed from the state shown in Fig. 10D, the schema version Sv is updated. Thus, the schema version is changed to Sv = 3. Consequently, in the state shown in Fig. 10E, a process for adding a container entry C to the immediately lower hierarchical level of the
10 container entry A is expressed by (3, A, +C). When this process is performed, a directory structure 105 shown in Fig. 10F is generated.

15 In the example shown in Figs. 10A to 10F, (1, A, +B), (2, A, +a), (2, A, -a), (2, A, -B), and (3, A, +C) are difference update information in individual states.

20 Figs. 11A, 11B, 11C, and 11D show another example of which a directory structure is changed. In the example shown in Figs. 10A to 10F, one process is performed at a time. However, in Figs. 11A to 11D, two processes are performed at a time. Fig. 11A shows a directory structure 106 of which only a container entry A is disposed in the immediately lower hierarchical level of the root entry (not shown). This state is
25 referred to as schema version Sv = 1. In the state shown in Fig. 11A, processes of (1, A, +B) and (1, A, +a) are successively performed. In other words, a

container entry B and a leaf entry a are added in the immediately lower hierarchical level of the container entry A. Thus, a directory structure 107 shown in Fig. 11B is generated. In this state, the hierarchical image of the container entries is changed. Thus, the schema version is changed to $Sv = 2$.

In the state shown in Fig. 11B, two processes of (2, A, -a) and (2, B, +b) are successively performed. In other words, a leaf entry a is deleted from the immediately lower hierarchical level of the container entry A. Thereafter, a leaf entry b is added to the immediately lower hierarchical level of the container entry B. Thus, a directory structure 108 shown in Fig. 11C is generated.

In the state shown in Fig. 11C, two processes of (2, B, +C) and (2, C, +c) are performed. In other words, a container entry C is added to the immediately lower hierarchical level of the container entry B. Thereafter, a leaf entry c is added to the immediately lower hierarchical level of the container entry C. In this case, since a leaf entry is added to an added container entry, the order of the processes cannot be changed. Thereafter, a directory structure 109 shown in Fig. 11D is generated. Since the hierarchical image of the container entry is changed, the schema version Sv is updated. Thus, the schema version is changed to $Sv = 3$.

In the example shown in Figs. 11A to 11D, (1, A, +B) and (1, A, +a), (2, A, -a) and (2, B, +b), and (2, B, +C) and (2, C, +c) are difference update information in individual stages. As was described above, when a plurality of processes are performed as one updating process of a directory structure, the order of processes should be considered.

Fig. 12 shows overall structure information corresponding to difference update information shown in Figs. 10A to 10F. The overall structure information is transmitted from the transmitting side replicater 12 at intervals of a predetermined time period T. In the following description, the time period T in which the overall structure information is transmitted is referred to as overall structure information notification period T. In Fig. 12, it is assumed that time passes from the left side to the right side. In addition, it is assumed that the intervals of changes of the directory structure (namely, structures 100, 101, 102, 103, and 104) are constant. In this example, the overall structure information notification period T corresponds to the interval of the structures 100 to 102 and the interval of the structures 102 to 104.

As was described above, in this example, the overall structure information is represented as difference information from the root entry to a directory structure at a particular time point. Thus,

as shown in Fig. 12, the overall structure information corresponding to the structure 102 can be expressed as (0, root, +A), (0, A, +B), and (0, A, +a) in the order of the processes. In other words, the individual processes are performed to the root entry in such an order. Individual entries are added and thereby the structure 102 is expressed. Likewise, in the structure 104, only the container entry A is disposed in the immediately lower hierarchical level of the root entry. Thus, the overall structure information at the time point corresponding to the structure 104 is expressed as (0, root, +A).

In the overall structure information notification period T, the schema version Sv is reset to 0. Thus, Sv in the overall structure information is always 0.

Fig. 13 shows overall structure information corresponding to the structures shown in Figs. 11A to 11D. In this example, the interval of the structures 106 to 109 corresponds to the overall structure information notification period T'. The overall structure information corresponding to the structure 109 can be represented by (0, root, +A), (0, A, +B), (0, B, +b), (0, B, +C), and (0, C, +c) in the order of processes.

The difference update information and the overall structure information of the directory

structure are not limited to the above-described examples. Instead, they may be changed corresponding to the applied system.

The contents of a leaf entry may be modified along with a deletion and an addition thereof from/to the immediately lower hierarchical level of a container entry. When the contents of a leaf entry are modified, the directory structure is not changed. In this case, difference update information is generated with for example a leaf entry name and a sequence of attribute names and attribute values that were modified. In this case, the difference update information is expressed as follow.

```
{  
    LeafEntryName,  
    Set of {AttributeName, AttributeValue}  
}
```

In the system according to the present invention, as was described above, difference update information and overall structure information are unidirectionally transmitted from the transmitting side 1 to the receiving side 3 through the broadcasting network 2. In addition, there are many receiving devices on the receiving side 3 against one transmitting side 1. In addition, the operating states of the individual receiving devices on the receiving side 3 differ. Thus, it is necessary to synchronize

directory information managed on the transmitting side 1 with directory information managed on the receiving side 3.

5 Next, a method for synchronizing directory information stored in the transmitting side server 11 on the transmitting side 1 with directory information stored in the receiving side server 16 on the receiving side 3 and managing the synchronization of the directory structure will be described.

10 First, with reference to Fig. 14, a method for managing the synchronization of container entries will be described. At step S1, the transmitting side client 10 changes the structure of the container hierarchy of a directory structure managed by the transmitting side server 11. For example, the
15 transmitting side client 10 performs a process for adding a new container entry and/or a leaf entry to the immediately lower hierarchical level of a particular container entry and a process for deleting a container
20 entry and/or a leaf entry from the immediately lower hierarchical level of a particular container entry.

25 At step S2, the transmitting side replicater 12 detects a change performed in the transmitting side server 11. Corresponding to the detected result, the transmitting side replicater 12 generates container structure update information Msg.1 corresponding to the change of the container hierarchical structure. The

generated container structure update information Msg.1
is broadcast to the broadcasting network 2. The same
contents of the container structure update information
Msg.1 are cyclically broadcast a predetermined number
of times.

At step S3, the container structure update
information Msg.1 that has been broadcast is received
by the receiving side replicater 17. The receiving
side replicater 17 changes the container hierarchy
structure managed with directory information stored in
the receiving side server 16 corresponding to the
received container structure update information Msg.1.
Thus, the structure of the container hierarchy of the
directory information on the transmitting side 1 is
synchronized with that on the receiving side 3.

The format of the container structure update
information Msg.1 is expressed as follows.

```
Container Structure Update Message {  
    MessageID,  
    Difference update information  
}
```

"MessageID" is identification information of the
message (container structure update information Msg.1).
For example, the "Message ID" is an integer that is
incremented by 1 whenever a message is generated.
"Difference update information" is difference update
information of the above described directory structure

corresponding to a change of the container hierarchy structure.

The process performed at step S2 of the flow chart shown in Fig. 14 will be described in detail with reference to a flow chart shown in Figs. 15A and 15B. All the process of the flow chart shown in Figs. 15A and 15B is performed by the transmitting side replicater 12. At step S100, an overall structure information notification period timer T_1 that counts the overall structure information notification period T is set to a multiple of a time period of a timer T_2 that is set at step S102. Thereafter, the overall structure information notification period timer T_1 is started. The detailed setting method of the overall structure information notification period timer T_1 will be described later.

After the overall structure information notification period timer T_1 is set, the flow advances to step S101. At step S101, all information of the hierarchical structure of the container entries of the transmitting side server 11 is read. The information of the hierarchical structure of the container entries is stored as a copy 1 to the recording medium or the storing medium such as a memory or a hard disk of the transmitting side replicater 12.

After the copy 1 is stored, the flow advances to step S102. At step S102, the timer T_2 is set to a

predetermined time period. Thereafter, the timer T_2 is started. Thereafter, the flow advances to step S103. At step S103, the overall structure information notification period timer T_1 and the timer T_2 count
5 respective set time periods. When one of the time periods that has been set to the overall structure information notification period timer T_1 and the timer T_2 elapses, the flow advances to step S104.

At step S104, it is determined whether or not
10 the predetermined time period of the overall structure information notification period timer T_1 has elapsed. When the determined result at step S104 is No (namely, the time period that had been set to the timer T_2 has elapsed), the flow advances to step S106.

15 On the other hand, when the determined result at step S104 is Yes (namely, the time period that had been set to the overall structure information notification period timer T_1 has elapsed), the flow advances to step S105. At step S105, the contents of
20 the copy 1 stored at step S101 are cleared. Thus, the contents of the copy 1 are only the root entry. Thereafter, the flow advances to step S106.

At step S106, all information of the
25 hierarchical structure of the container entries of the transmitting side server 11 is read. The hierarchical structure of the container entries that are read is stored as a copy 2 to the recording medium or the

storing medium such as a memory or a hard disk of the transmitting side replicater 12.

At step S107, the copy 1 is compared with the copy 2 stored at step S106. Thereafter, the flow advances to step S108. At step S108, it is determined whether there is a difference between the copy 1 and the copy 2. When the determined result at step S108 is No (namely, there is no difference between the copy 1 and the copy 2), the flow returns to step S100. At step S100, the overall structure information notification period timer T_1 is set and the process after step S100 is repeated.

In the case that the time period that had been set to the overall structure information notification period timer T_1 has elapsed (the determined result at step S104 is Yes) and the contents of the copy 1 has been cleared at step S105, the copy 1 is compared with the copy 2 at step S107. Thus, the difference of the hierarchical structure of the container entries stored as the copy 2 against the root entry is obtained.

On the other hand, when the determined result at step S108 is Yes (namely, there is a difference between the copy 1 and the copy 2), the flow advances to step S109. At step S109, corresponding to the difference between the copy 1 and the copy 2, difference update information is generated. Thus,

container structure update information Msg.1 containing the difference update information is generated. The container structure update information Msg.1 is transmitted to the broadcasting network 2. Thus, the container structure update information Msg.1 is broadcast through the broadcasting network 2. The broadcast container structure update information Msg.1 is received by the receiving side replicater 17.

When the container structure update information Msg.1 is broadcast at step S109, the flow advances to step S110. At step S110, the contents of the copy 1 are substituted with the contents of the copy 2. Thereafter, the flow returns to step S100.

Next, the process of step S3 of the flow chart shown in Fig. 14 will be described in detail with reference to a flow chart shown in Fig. 16. All the process of the flow chart shown in Fig. 16 is performed by the receiving side replicater 17. At step S20, the container structure update information Msg.1 that has been broadcast through the broadcasting network 2 by the transmitting side replicater 12 is received by the receiving side replicater 17.

At step S21, it is determined whether or not the container structure update information Msg.1 has been received first time at step S20. When the determined result at step S21 is Yes (namely, the container structure update information Msg.1 has been

received first time), the flow advances to step S23.
At step S23, the message ID contained in the container
structure update information Msg.1 is stored as a copy
3 to the recording medium or the storing medium such as
5 a memory or a hard disk of the receiving side
replicater 17.

Thereafter, the flow advances to step S24.
At step S24, corresponding to the contents of the
received container structure update information Msg.1
10 (namely, the difference update information contained in
the container structure update information Msg.1), the
directory information managed by the receiving side
server 16 is updated. The structure of the container
hierarchy represented by the directory information is
15 changed. Thereafter, the flow returns to step S20.

On the other hand, when the determined result
at step S21 is No (namely, the container structure
update information Msg.1 has been received at step S20
not first time), the flow advances to step S22. At
20 step S22, it is determined whether or not the message
ID contained in the received container structure update
information Msg.1 is the same as the message ID stored
as the copy 3 at step S23. When the determined result
at step S22 is Yes (namely, they are the same), the
25 flow returns to step S20.

On the other hand, when the determined result
at step S22 is No (namely, they are not the same), the

flow advances to step S23. At step S23, as was described above, the message ID is stored as the copy 3 to the storing medium. In this case, the message ID that has been received and stored is overwritten with the message ID that has been newly received.

Thereafter, the flow advances to step S24. At step S24, corresponding to the received container structure update information Msg.1, the contents of the container entry hierarchy of the receiving side server 16 are changed.

Next, with reference to a flow chart shown in Fig. 17, a method for managing the synchronization of leaf entries will be described. At step S30, the transmitting side client 10 changes a leaf entry in the immediately lower hierarchical level of a particular container entry of a directory structure managed by the transmitting side server 11. For example, the transmitting side client 10 performs a process for adding a new leaf entry to the immediately lower hierarchical level of a particular container entry, a process for deleting a leaf entry from the immediately lower hierarchical level of a particular container entry, or a process for modifying a leaf entry of the immediately lower hierarchical level of a particular container entry.

At step S31, the transmitting side replicater 12 detects a change of a leaf entry in the immediately

lower hierarchical level of a particular container entry of the transmitting side server 11. Corresponding to the detected result, leaf update information Msg.x1 due to the change of the leaf entry in the immediately lower hierarchical level of the particular container entry is generated. The generated leaf update information Msg.x1 is cyclically broadcast to a plurality of receiving side replicaters 17 through the broadcasting network 2.

At step S32, the leaf update information Msg.x1 that has been broadcast is received by the receiving side replicaters 17. Each receiving side replicater 17 changes a relevant leaf entry managed with the directory information stored in the receiving side server 16 corresponding to the received leaf update information Msg.x1. Thus, a leaf entry of the directory information on the transmitting side 1 is synchronized with a leaf entry of the directory information on the receiving side 3.

The format of the leaf update information Msg.x1 is expressed as follows.

```
Leaf Entry Update Message {  
    MessageID,  
    Difference update information  
}
```

"MessageID" is identification information of the message (leaf update information Msg.x1). For example,

the "Message ID" is an integer that is incremented by 1 whenever a message is generated. "Difference update information" is difference update information of the above-described directory structure.

5 Next, the process of step S31 of the flow chart shown in Fig. 17 will be described in detail with reference to a flow chart shown in Figs. 18A and 18B. All the process of the flow chart shown in Fig. 17 is performed by the transmitting side replicater 12. The process of the flow chart shown in Figs. 18A and 18B is performed by the transmitting side replicater 12 for all container entries of the directory structure managed by the transmitting side server 11. At step S120, an overall structure information notification period timer T_1' that counts the overall structure information notification period T is set to a multiple of a time period of a timer T_2 that is set at step S122. Thereafter, the timer T_1 is started. As will be described later, different time periods can be set to the overall structure information notification period timer T_1' and the above-described overall structure information notification period timer T_1 .

20 After the timer T_1 is set, the flow advances to step S121. At step S121, leaf entry names in the immediately lower hierarchical level of a particular container entry (referred to as container entry A) of the transmitting side server 11 are read. The leaf

entry names that have been read are stored as a copy 4 to the recording medium or the storing medium such as a memory or a hard disk of the transmitting side replicater 12.

5 After the copy 4 is stored, the flow advances to step S122. At step S122, the timer T_2 is set to a predetermined time period. Thereafter, the flow advances to step S123. At step S123, the overall structure information notification period timer T_1' and the timer T_2 count the respective time periods that have been set at step S120. Thereafter, the flow advances to step S124. At step S124, it is determined whether or not the time period that had been set to the overall structure information notification period timer T_1 has elapsed.

10
15 When the determined result at step S124 is No (namely, the time period that has been set to the timer T_2 has elapsed), the flow advances to step S126.

20 On the other hand, when the determined result at step S124 is Yes (namely, the time period that had been set to the overall structure information notification period timer T_1' has elapsed), the flow advances to step S125. At step S125, the contents of the copy 4 stored at step S121 are cleared. Since the contents of the copy 4 are cleared, the copy 4 contains only the above-described container entry A. Thereafter, the flow advances to step S126.

At step S126, all leaf entry names in the immediately lower hierarchical level of a particular container entry (in this example, the container entry A) of the transmitting side server 11 are read. The leaf entry names that have been read are stored as a copy 5 to the recording medium or the storing medium such as a memory or a hard disk of the transmitting side replicater 12.

At step S127, the copy 4 is compared with the copy 5 stored at step S126. Thereafter, the flow advances to step S128. At step S128, it is determined whether or not there is a difference between the copy 4 and the copy 5. When the determined result at step S128 is No (namely, there is no difference), the flow returns to step S120. At step S120, the overall structure information notification period timer T_1' is set. Thereafter, the process after step S120 is repeated.

On the other hand, when the determined result at step S128 is Yes (namely, there is a difference between the copy 4 and the copy 5), the flow advances to step S129. At step S129, corresponding to the difference between the copy 4 and the copy 5, difference update information is generated. Thus, leaf update information Msg.x1 containing the difference update information is generated. The generated leaf update information Msg.x1 is transmitted to the

broadcasting network 2. The leaf update information
Msg.x1 is broadcast through the broadcasting network 2.
The broadcast leaf update information Msg.x1 is
received by a plurality of receiving side replicaters
17.

When the leaf update information Msg.x1 is
broadcast at step S129, the flow advances to step S130.
At step S130, the contents of the copy 4 are
overwritten with the contents of the copy 5.
Thereafter, the flow returns to step S120.

In the case that the time period that has
been set to the overall structure information
notification period timer T_1' has elapsed at step S124
and the contents of the copy 4 has been cleared at step
S125, corresponding to the difference update
information generated at step S129, a new leaf entry is
added to the container entry A of which information in
the immediately lower hierarchical level thereof has
been cleared.

Next, the process of step S32 of the flow
chart shown in Fig. 17 will be described in detail with
reference to a flow chart shown in Fig. 19. All the
process of the flow chart shown in Fig. 19 is performed
by the receiving side replicater 17. At step S50, leaf
update information Msg.x1 that has been broadcast by
the transmitting side replicater 12 through the
broadcasting network 2 is received by the receiving

side replicater 17.

At step S51, it is determined whether or not the leaf update information Msg.x1 has been received at step S50 first time. When the determined result at
5 step S51 is Yes (namely, the leaf update information Msg.x1 has been received first time), the flow advances to step S53. At step S53, the message ID of the received leaf update information Msg.x1 is stored as a copy 6 to the recording medium or the storing medium
10 such as a memory or a hard disk of the receiving side replicater 17.

Thereafter, the flow advances to step S54. At step S54, corresponding to the contents of the received leaf update information Msg.x1 (namely, the
15 difference update information contained in the leaf update information Msg.x1), a relevant leaf entry of the directory information managed by the receiving side server 16 is changed. Thereafter, the flow returns to step S50.

20 On the other hand, when the determined result at step S51 is No (namely, the leaf update information Msg.x1 has been received not first time), the flow advances to step S52. At step S52, it is determined whether or not the message ID contained in the received
25 leaf update information Msg.x1 is the same as the message ID stored as the copy 6 at step S53. When the determined result at step S52 is Yes (namely, they are

the same), the flow returns to step S50.

On the other hand, when the determined result at step S52 is No (namely, they are not the same), the flow advances to step S53. At step S53, as was described above, the message ID is stored as a copy 6 to the storing medium. In this case, the message ID that has been received and stored is overwritten with the message ID that has been newly received.

Thereafter, the flow advances to step S54. At step S54, corresponding to the received leaf update information Msg.x1, a relevant left entry of the receiving side server 16 is changed.

At step S31 of the flow chart shown in Fig. 17, the leaf update information Msg.x1 for all the container entries of the directory structure managed by the transmitting side 1 is broadcast. Thus, it is supposed that the data amount of the leaf update information Msg.x1 is huge. Thus, as was described as a problem mentioned in the section of the related art reference, when the receiving side 3 receives all the leaf update information Msg.x1 and performs the process shown in Fig. 19 for it, the receiving side 3 is adversely loaded. To prevent that, the receiving side 3 should effectively filter only the leaf update information Msg.x1 of leaf entries in the immediately lower hierarchical level of container entries that are frequently inquired from that for all the container

entries that have been broadcast.

For example, it is assumed that the receiving side replicater 17 is used along with for example a set top box (STB) that is connected to a television receiver or the like and that has a limited processing capability and storing capacity (in other words, the receiving side replicater 17 is used in an environment of an insufficient computer resource). In this case, the receiving data amount of the leaf update information Msg.x1 that is broadcast is limited. Thus, it is necessary to select the received leaf update information Msg.x1 and supply the selected data to the receiving side replicater 17 so as to reduce the storage cost and the message processing cost. In other words, the cost for storing and processing unnecessary data should be reduced. In particular, as the directory service is becoming common and the directory structure managed by the transmitting side server 11 is becoming huge, the selecting process for the leaf update information Msg.x1 is becoming important.

Next, the filtering process for the leaf update information Msg.x1 will be described. In addition, an effective filtering method according to the present invention will be described. The transmitting side replicater 12 adds a filtering mask to leaf update information Msg.x1 to be broadcast. With the filtering mask, the receiving side replicater

17 performs a filtering process. A mask schema structure for interpreting a filtering mask and a method for causing the transmitting side replicater 12 to notify the receiving side replicater 17 of a mask schema structure will be described later.

The structure of a message (Msg.xl') of which a filtering mask has been added to leaf update information Msg.xl is defined as follows. The above-described leaf update information Msg.xl is substituted with the leaf update information Msg.xl'. In other words, the leaf update information Msg.xl' is defined as follows:

```
Leaf Entry Update Message {  
    MessageID,  
    FilteringMask,  
    Difference update information  
}
```

As with the above-described leaf update information Msg.xl, "MessageID" is an integer as identification information of the message (leaf update information Msg.xl'). For example, whenever a message is generated, the "MessageID" is incremented by 1. "Difference update information" is information that represents a process such as addition, deletion, or attribute change of a leaf entry in the immediately lower hierarchical level of a container entry designated by the filtering mask.

The structure of "FilteringMask" (filtering mask) is defined as follows.

```
FilteringMask {  
    MaskSchema Version,  
    Mask Value  
}
```

"MaskSchema Version" is equivalent to a message ID of the above-described container structure update information Msg.1. Whenever a filtering mask is generated, the "MaskSchema Version" is incremented by 1. "Mask Value" is the value of a mask represented as for example a bit string or on the order of bytes.

The structure of the mask value is defined by a mask schema corresponding to a mask schema version. The mask schema will be described later. The transmitting side replicater 12 notifies the receiving side replicater 17 of the mask schema with another message that will be described later.

Next, the method for assigning a mask value will be described. According to the embodiment, each of container entries in the immediately lower hierarchical level of a particular container entry is identified with a bit string composed of a predetermined number of bits. With reference to a mask value contained in the received leaf update information Msg.x1', the receiving side replicater 17 performs a filtering process so as to selectively extract desired

leaf update information Msg.x1'.

The bit array structure of the mask value of a filtering mask is designated corresponding to the hierarchical structure of container entries. For example, as shown in Fig. 20A, corresponding to the assigning method for entry names described with reference to Fig. 4, to identify entries X.A, X.B, X.C, X.D, and X.E in the immediately lower hierarchical level of a particular container entry X, three-bit mask values (000), (001), (010), (011), and (100) are assigned. In Fig. 20, "... " represents that there is a container entry in the immediately higher hierarchical level of the current entry.

When an entry is added or deleted to/from a container entry in the immediately lower hierarchical level of a container entry X, a process of a flow chart shown in Fig. 21 is performed. A mask value is assigned corresponding to an addition or a deletion of a container entry. In the following description, a container hierarchy in the state before a container entry is added or deleted is referred to as pre-update container hierarchy. In this example, it is assumed that the number of mask digits M' of the pre-update container hierarchy has been stored in for example the memory of the transmitting side replicater 12.

At step S60, the transmitting side replicater 12 obtains the number of container entries N in the

immediately lower hierarchical level of a target container entry. With reference to the list of the container entries in the immediately lower hierarchical level of the target container entry, the number of container entries N is obtained. Thereafter, the flow advances to step S61. At step S61, the number of bits M that can uniquely identify N elements is selected. Thus, the number of mask digits is designated to M. In the example shown in Fig. 20A, since the container entry X has five container entries in the immediately lower hierarchical level thereof, [3] bits that can uniquely identify the five container entries are designated as the number of mask digits.

Thereafter, the flow advances to step S62. At step S62, it is determined whether or not the number of bits M designated at step S61 is the same as the number of mask digits M' designated to the pre-update container hierarchy. When the determined result at step S62 is Yes (namely, the number of mask digits M is the same as the number of mask digits M'), the flow advances to step S63.

At step S63, container entries of the post-update container hierarchy corresponding to those of the pre-update container hierarchy are designated the same mask value. Thereafter, the flow advances to step S64. At step S64, when there is a container entry of the pre-update container hierarchy that does not

correspond to that of the post-update container hierarchy, the container entry is assigned a unique mask value that is not used for mask values of the other container entries of the same container hierarchy.

On the other hand, when the determined result at step S62 is No (namely, the number of mask digits M is not the same as the number of mask digits M'), the flow advances to step S65. At step S65, unique mask values are assigned to all the container entries of the container hierarchy.

Now, consider that case that a new container entry "... X.F" is added to the state shown in Fig. 20A and thereby a container hierarchy shown in Fig. 20B is generated. In this case, since the number of container entries N in the immediately lower hierarchical level of the container entry "... X" is 6. To uniquely identify the six container entries, three bits are required. Thus, the number of mask digits M of the container hierarchy in the immediately lower hierarchical level of the post-update container entry "... X" is $M = 3$. Since the number of mask digits M' of the pre-update container hierarchy is $M' = 3$, the number of mask digits M' is the same as the number of mask digits M . Thus, mask values of entries of the pre-update container hierarchy are assigned to the container entries "... X.A", "... X.B", "... X.C",

"... X.D", and "... X.E" shown in Fig. 20B (at step S63). On the other hand, the container entry "... X.F" that has been newly added is designated a unique mask value (101) that is different from other mask values of the other container entries of the same container hierarchy (at step S64).

Next, consider the case that the container entry "... X.C" is deleted from the state shown in Fig. 20A and thereby a container hierarchy shown in Fig. 20C is generated. In this case, the number of container entries N in the immediately lower hierarchical level of the container entry "... X" is $N = 4$. Thus, with two bits as the number of mask digits M , these container entries can be uniquely identified. Consequently, the number of mask digits M of the post-update container hierarchy is $M = 2$. On the other hand, the number of mask digits M' of the pre-update container hierarchy is $M' = 3$. Thus, the number of mask digits M' of the pre-update container hierarchy is not the same as the number of mask digits M of the post-update container hierarchy. In this case, the flow advances to step S65. At step S65, all the entries of the hierarchy are assigned new mask values with the number of mask digits $M = 2$.

Next, consider that case that a new container entry "... X.G" is added to the state shown in Fig. 20C and thereby a state shown in Fig. 20 is generated.

In this case, the number of container entries N in the immediately lower hierarchical level of the container entry "... X" is $N = 5$. To uniquely identify these container entries, the number of mask digits should be

5 $M = 3$. However, the number of mask digits M of the post-update container hierarchy is not the same as the number of mask digits M' of the pre-update container hierarchy. Thus, in this case, new mask values are assigned to all the container entries in the

10 immediately lower hierarchical level of the container entry "... X" at step S65.

Mask values are bit-assigned from the highest hierarchical level of the directory structure. On the other hand, according to the embodiment of the present

15 invention, as was described above, the number of mask digits depends on the number of entries of the same hierarchical level. In addition, when an entry is deleted or added, the number of entries of the container hierarchy changes. Thus, the number of mask

20 digits changes. Consequently, an information mechanism for determining the relation between bits of the bit string that represent mask values and container entries (or container hierarchy) and for interpreting mask values is required.

25 According to the embodiment of the present invention, a mask schema (MaskSchema) is defined as follows:

```

MaskSchema {
    MaskSchema Version,
    TotalMaskLength,
    Set of ContainerEntryMaskSchema
5      }

```

"MaskSchema Version" (mask schema version) is equivalent to the message ID of the above-described container structure update information Msg.1. Whenever a filtering mask is generated, the "MaskSchema Version" is incremented by 1. "TotalMaskLength" (total mask length) represents the total bit length of all mask values of the overall container hierarchy. In other words, the total mask length corresponds to the number of bits required to represent all hierarchical levels of the directory structure. "Set of ContainerEntryMaskSchema" (set of container entry mask schema) represents an array of "ContainerEntryMaskSchema" (container entry mask schema) that will be described later.

The above-described container entry schema defines a filtering mask corresponding to a particular container entry. In other words, the container entry mask schema is defined as follows:

```

ContainerEntryMaskSchema {
25     ContainerEntryName,
        OffsetLength,
        MaskLength,

```

AssignedMaskValue

}

"ContainerEntryName" (container entry name) is a character string that represents the entry name of a target container entry. "OffsetLength" (offset length) is an offset value from the first bit of all mask values of a filtering mask of the container entry. "MaskLength" (mask length) represents the number of digits (bit length) of a mask value.

"AssignedMaskValue" (assigned mask value) is a mask value as a bit string assigned to an object container entry.

Next, with reference to Figs. 22A and 22B, a container entry mask schema that is coded will be described. Fig. 22A corresponds to Fig. 20A. Referring to Fig. 22A, there are five container entries " ... X.A", " ... X.B", " ... X.C", " ... X.D", and " ... X.E" in the immediately lower hierarchical level of a particular container entry " ... X". Mask values with a mask length of three digits are assigned to the five container entries. In this example, it is assumed that these five container entries do not have other entries in the immediately lower hierarchical level thereof.

Fig. 22B shows an example of the mask value of a container entry " ... X.C". In this example, since the offset length is 77 bits, it is clear that

the mask value assigned to the container entry " ...
X.C" with a mask length of three bits is three bits
starting from 78-th bit of the mask value the container
entry " ... X.C". The mask value of the 77 bits
5 contained in the offset length is an assigned mask
value corresponding to a container entry in the
immediately upper hierarchical level of the container
entry " ... X.C".

10 In such a manner, the position of a mask
value assigned to a target container entry is defined
and a container entry mask schema is coded.

Next, a more practical example of a container
entry mask schema will be described. A container entry
mask schema corresponding to the above-described
15 container entry " ... X.C" is defined as follows:

```
ContainerEntryMaskSchema {  
    " ... X.C", (ContainerEntryName)  
    77, (OffsetLength)  
    3, (MaskLength)  
20    010 (AssignedMaskValue)  
    }
```

Characters in parentheses are just for comments and
omissible.

25 A container entry schema corresponding to the
container entry " ... X.D" shown in Fig. 22A is for
example defined as follows:

```
ContainerEntryMaskSchema {
```

```
" ... X.D",
```

```
77,
```

```
3,
```

```
011
```

```
5      }
```

Assuming that the mask schema version is 498 and that the total mask length is 134 bits, the mask schema is defined as follows:

```
MaskSchema {
```

```
10      498, (MaskSchema Version)
```

```
134, (TotalMaskLength)
```

```
...
```

```
ContainerEntryMaskSchema {
```

```
" ... X.C",
```

```
15      77,
```

```
3,
```

```
010
```

```
}
```

```
ContainerEntryMaskSchema {
```

```
20      " .... X.D,
```

```
77,
```

```
3,
```

```
011
```

```
}
```

```
25      ...
```

```
}
```

In the above example, the container entry mask schemas

of the container entries " ... X.C" and "... X.D" are contained in a mask schema. In reality, another container entry mask schema is contained in " ... ". As is clear from this example, a mask schema contains
5 container entry mask schemas of all container entries of one directory structure.

In this example, although the total mask length is 134 bits, the offset value and the mask length of the container entry mask schema of the
10 container entries " ... X.C" and " ... X.D" are 77 bits and 3 bits, respectively. Thus, the total bit length is 80 bits. This means that the container entries " ... X.C" and " ... X.D" contain lower container hierarchies.

15 The filtering mask corresponding to the container entry " ... X.C" of the above-described mask schema is coded as follows:

```
FilteringMask {  
    498, (MaskSchema Version)  
20    ..... 010 (Mask Value)  
    }
```

Mask values other than "011" are filled with bits of mask values assigned to container entries of other hierarchical levels.

25 Likewise, the filtering mask corresponding to the container entry " ... X.D" is coded as follows:

```
FilteringMask {
```


498, (MaskSchema Version)

..... 011 (Mask Value)

}

5 The transmitting side replicater 12 monitors
the transmitting side server 11, detects a change of
the hierarchical structure of container entries, and
changes the above-described mask schema. Thus, when
the receiving side 3 properly performs a filtering
process, the transmitting side replicater 12 should
10 notify the receiving side replicater 17 of the changed
mask schema along with difference update information
corresponding to the change of the hierarchical
structure.

15 According to the present invention, to allow
the transmitting side replicater 12 to notify the
receiving side replicater 17 of the mask schema, the
mask schema structure is added to the structure of the
above-described container structure update information
Msg.1. Container structure update information Msg.1'
20 to which a mask schema structure is added is defined as
follows:

```
Container Structure Update Message {  
    MessageID,  
    difference update information,  
25     MaskSchema  
}
```

Whenever the structure of the container hierarchy is

changed, the mask schema may be changed. Thus, the container structure update information Msg.1' is generated corresponding to a change of the structure of the container hierarchy. "MessageID" (message ID) is an integer that is incremented by 1 whenever the container structure update information Msg.1' is generated. In the following description, the above-described container structure update information Msg.1 is substituted with the container structure update information Msg.1'.

The transmitting side replicater 12 generates leaf update information Msg.x1' that is a message to which a filtering mask corresponding to the container hierarchy has been added at step S46 of the flow chart shown in Figs. 18A and 18B. The transmitting side replicater 12 broadcasts the generated leaf update information Msg.x1' to the receiving side replicater 17. Before the receiving side replicater 17 performs a filtering process for the leaf update information Msg.x1', the receiving side 3 should have designated of a target portion of the container hierarchy that the receiving side client 15 requires.

According to the embodiment of the present invention, the receiving side replicater 17 generates a target mask list that lists masks for processing a target container hierarchy.

Next, with reference to Figs. 23A and 23B, a

target mask will be described. First of all, a directory structure as shown in Fig. 23A is assumed. It is assumed that the directory structure shown in Fig. 23A is composed of only container entries except
5 for a root entry at the highest hierarchical level. In Fig. 23A, a single square represents a container entry, whereas a concentric square represents a container entry that the user designates for the filtering process performed by the receiving side client 15. In
10 Fig. 23A, a numeral in each entry represents a mask value assigned thereto.

As shown in Fig. 23A, container entries designated by the user are assigned masks 1 to 5 for the filtering process performed by the receiving side
15 client 15. In the directory structure, the mask values of the total mask length of the masks 1, 2, 3, 4, and 5 are "000", "0010", "010", "1000", and "10010", respectively.

Fig. 23B shows an example of a target mask
20 list that lists the designated masks. The target mask list is composed of a schema version that identifies the directory structure and a list of mask values designated by the receiving side client 15. In other words, the target mask list is a list valid for only
25 the directory structure represented by the schema version.

Fig. 24 is a flow chart showing a process for

generating a target mask list. This process is
executed by the receiving side replicater 17. At step
S70, the receiving side replicater 17 receives
container structure update information Msg.1'. At step
5 S71, it is determined whether or not the container
structure update information Msg.1' has been received
first time. When the determined result at step S71 is
Yes (namely, the container structure update information
Msg.1' has been received first time), the flow advances
10 to step S73.

At step S73, the message ID contained in the
received container structure update information Msg.1'
is stored as a copy 7 to the storing medium or the
recording medium such as a memory or a hard disk of the
15 receiving side replicater 17.

At step S74, corresponding to the contents of
the received container structure update information
Msg.1', a container hierarchy is generated. The
receiving side replicater 17 notifies the receiving
20 side client 15 of information that represents the
generated container hierarchy so as to prompt the
receiving side client 15 for selecting a container
entry to be designated. For example, the receiving
side client 15 causes a predetermined displaying means
25 to display information corresponding to the supplied
container hierarchy. The user selects a required
container entry with reference to the information

displayed on the displaying means. The selected container entry information is supplied from the receiving side client 15 to the receiving side replicater 17.

5 A container entry may be designated by other than the user. In other words, the receiving side client 15 may store container entry information inquired by the user, learn the user's favorites corresponding to the stored information, and
10 automatically select a container entry corresponding to the learnt result. Alternatively, a container entry may be designated in a combination of user's direct selection and automatic selection using such a learning process.

15 After a container entry has been selected at step S74, the flow advances to step S75. At step S75, a filtering mask corresponding to the selected container hierarchy is designated. A list of the designated filtering masks is stored as a target mask
20 list to the recording medium or the storing medium such as a memory or a hard disk of the receiving side replicater 17.

25 On the other hand, when the determined result at step S71 is No (namely, the container structure update information Msg.1' has been received not first time), the flow advances to step S72. At step S72, it is determined whether or not the message ID contained

in the received container structure update information
Msg.1' is the same as the message ID stored as copy 7
in the storing medium at step S73.

When the determined result at step S72 is Yes
(namely, they are the same), the flow returns to step
S70. On the other hand, when the determined result at
step S72 is No (namely, they are not the same), the
flow advances to step S74. At step S74, the message ID
contained in the newly received container structure
update information Msg.1' is stored to the storing
medium instead of the old message ID. Corresponding to
the container structure update information Msg.1' that
has been newly received, the following process is
performed.

Fig. 25 is a flow chart showing a process for
selectively receiving leaf update information Msg.x1'
(that has been broadcast) corresponding to a target
mask list generated in the process of the flow chart
shown in Fig. 24. The receiving side replicater 17
selectively receives leaf update information Msg.x1'
having a filtering mask listed in the target mask list
from the leaf update information Msg.x1' that has been
broadcast through the broadcasting network 2.
Corresponding to the leaf update information Msg.x1'
that has been selectively received, the process of step
S32 of the flow chart shown in Fig. 17 is executed.

In Fig. 25, at step S80, the receiving side

replicater 17 receives leaf update information Msg.xl' that has been broadcast through the broadcasting network 2. The receiving side replicater 17 references the target mask list stored in the storing medium and determines whether or not the filtering mask contained in the received leaf update information Msg.xl' is contained in the target mask list. When the determined result at step S81 is No (namely, the filtering mask is not contained in the target mask list), the flow returns to step S80.

On the other hand, when the determined result at step S81 is Yes (namely, the filtering mask is contained in the target mask list), the flow advances to step S82. At step S82, it is determined whether or not the leaf update information Msg.xl' has been received first time. When the determined result at step S82 is Yes (namely, the leaf update information Msg.xl' has been received first time), the flow advances to step S84. The message ID contained in the received leaf update information Msg.xl' is stored as a copy 8 to the recording medium or the storing medium such as a memory or a hard disk of the receiving side replicater 17. Thereafter, the flow advances to step S85. At step S85, the received leaf update information Msg.xl' is selected as a target of the process of the receiving side replicater 17.

On the other hand, when the determined result

at step S82 is No (namely, the leaf update information
Msg.x1' has been received not first time), the flow
advances to step S83. At step S83, it is determined
whether or not the message ID contained in the received
leaf update information Msg.x1' is the same as the
message ID stored as copy 8 to the storing medium at
step S84.

When the determined result at step S83 is Yes
(namely, they are the same), the flow returns to step
S80. On the other hand, when the determined result at
step S83 is No (namely, they are not the same), the
flow advances to step S84. At step S84, the message ID
contained in the received leaf update information
Msg.x1' is stored to the storing medium instead of the
old message ID. Corresponding to the newly received
leaf update information Msg.x1', the next process is
performed.

As described above, only a user's favorite
portion of the directory structure managed by the
transmitting side server 11 can be stored and updated
by the receiving side server 16. Thus, the storing
medium that stores the directory structure can be
effectively used in the receiving side server 16. In
addition, the storage cost of the directory structure
in the receiving side server 16 can be suppressed.
Moreover, the processing efficiency of the receiving
side client 15 corresponding to a search request for

contents data stored in the receiving side server 16 can be remarkably improved.

In the above-described embodiment, the mask length of a mask assigned to each container entry is variable. However, the present invention is not limited to such an example. According to the present invention, the mask length may be a fixed length (for example on the order of bytes).

Next, the setting methods of the above-described overall structure information notification period timers T_1 and T_1' will be described. As was described above, different time periods may be set to the overall structure information notification period timer T_1 used for deciding the timing of the broadcast of the container structure update information $\text{Msg}.1'$ and to the overall structure information notification period timer T_1' used for deciding the timing of the broadcast of the leaf update information $\text{Msg}.x1'$. In addition, the time periods that are set to the timers T_1 and T_1' can be changed at any time point.

According to the embodiment of the present invention, the transmitting side 1 (transmitting side replicater 12) monitors the operating state of the receiving side 3 (receiving side replicater 17) in a predetermined manner and designates time periods to be set to the timers T_1 and T_1' corresponding to the monitored result. The timers T_1 and T_1' are set to

different time periods that vary in each time zone corresponding to the operating state of the receiving side 3.

Fig. 26 is a flow chart showing an example of a process for setting time periods of the overall structure information notification period timers T_1 and T_1' . At step S140, the receiving side replicater 17 monitors the local operating state. In other words, the receiving side replicater 17 monitors the operating state at intervals of a predetermined time period. The chronological fluctuation of the operating state is stored to the recording medium or the storing medium such as a memory or a hard disk of the receiving side replicater 17. The operating state is obtained by cumulating the time period of the receiving process for two types of update messages in each time zone of the receiving side replicater 17.

The receiving side replicater 17 notifies the transmitting side replicater 12 of data that represents the operating state of the receiving side replicater 17 at step S141. The operating state information is transmitted from the receiving side replicater 17 to the transmitting side replicater 12 through the bidirectional network 4.

The transmitting side replicater 12 receives the operating state information of the receiving side replicater 17 transmitted through the bidirectional

network 4 and performs a predetermined statistic process corresponding to the received operating state information (at step S142). As an example, the statistic process is performed by cumulating the operating state information transmitted from a plurality of receiving side replicaters 17 at intervals of a predetermined time period. As the result of the statistic process, the statistic information of the chronological variation of the operating state of the receiving side replicater 17 is obtained. The obtained statistic information of the chronological variation is stored to the transmitting side replicater 12.

At step S143, the transmitting side replicater 12 sets the overall structure information notification period timers T_1 and T_1' corresponding to the result of the statistic process performed at step S142.

Now, it is assumed that the result of the statistic process performed at step S142 is a chronological variation as shown in Fig. 27. In Fig. 27, the horizontal axis represents time of one day, whereas the vertical axis represents the operating ratio of the receiving side replicater 17. The area b represents a time zone of which the operating ratio of the receiving side replicater 17 is high. The areas a and c represent time zones of which the operating ratio of the receiving side replicater 17 is relatively low.

Corresponding to the statistic results shown in Fig. 27, the overall structure information notification period timers T_1 and T_1' are set for each of the areas a, b, and c. The operating ratio of the receiving side replicater 17 in the area b is higher than that in the areas a and c. It is considered that the access ratio of the receiving side 3 against the broadcast from the transmitting side 1 is high. Thus, even if the timers T_1 and T_1' are set to large time periods and thereby the broadcast period of the difference update information is long, the probability of which the receiving side 3 drops the broadcast difference update information in the area a is lower than that of the areas a and c.

Thus, as an example, in the area b of which the operating ratio of the receiving side replicater 17 is high, long time periods are set to the timers T_1 and T_1' so as to prolong the broadcast intervals of the difference update information. In the areas a and c, short time periods are set to the timers T_1 and T_1' so as to shorten the broadcast intervals of the difference update information. Thus, according to the embodiment of the present invention, the intervals of the notification of the overall structure information that uses many communication resources can be dynamically varied corresponding to the operating ratio of the receiving side 3. Thus, the notification of the

difference update information of the directory structure can be effectively performed.

According to the above-described embodiment, the statistic data of the operating ratio of the receiving side replicater is obtained corresponding to time zones of one day. However, the present invention is not limited to such an example. For example, the statistic data of the variation information of the operating ratio may be obtained at intervals of one week, one month, or one year. Alternatively, the statistic process for the variation of the operating ratio can be performed in a combination of a plurality of intervals.

As was described above, according to the present invention, the directory structure managed on the transmitting side is broadcast with difference update information of container entries and leaf entries corresponding to a change of the structure. In addition, overall structure information that allows the overall directory structure to be reproduced is transmitted at intervals of a predetermined time period. The intervals of the notification of the overall structure information that uses many communication resources are dynamically varied corresponding to the operating ratio on the receiving side. Thus, the notification of the difference update information of the directory structure can be

effectively performed.

In addition, according to the present invention, a directory structure managed on the transmitting side is broadcast with difference update information of container entries and leaf entries 5 corresponding to a change of the structure. In addition, overall structure information that allows the overall hierarchical structure to be reproduced is transmitted at intervals of a predetermined time period. Thus, the receiving side can synchronize with 10 the transmitting side for a portion of the directory structure desired on the receiving side. In addition, the transmitting side can synchronize with a plurality of receiving sides.

Although the present invention has been shown and described with respect to a best mode embodiment thereof, it should be understood by those skilled in 15 the art that the foregoing and various other changes, omissions, and additions in the form and detail thereof may be made therein without departing from the spirit 20 and scope of the present invention.